SW Engineering CSC648/848

Pawster

Dog meeting app

Section 01 Team 5

Milestone 2

March 18, 2022

Name:	**Role:**	**Progress:**
Sina Pourdehmobed	Group Leader	- Functional requirements - Planning - Completed
John Paul Apolinar	Backend Lead	- Key Risks - Completed
Jonathan Chen	Frontend Lead	- Data Definitions V2 - Completed
Haozhan Li	Scrum Master	- Project management - Completed
Umar Rama	Backend Lead	- High level Architecture, Database Organization - Completed - High Level UML Diagrams - Completed
Henry Lam	Github Master	- UI Mockups and Storyboards (high level only)- Completed

Data Definitions V2
Functional Requirements V2
UI Mockups and Storyboards (high level only)
High level Architecture, Database Organization
High level UML Diagrams
Identify actual key risks for your project at this time
Project Management

Green= Completed
Red= In progress

1. Data Definitions V2

Jonathan Chen

The following definitions are key terms in relation to software components and database elements that will be incorporated and implemented into the structural design of our website. The following section will continue to be expanded upon throughout later milestones with a detailed synopsis for each key term as well as additional terms that will be implemented/utilized in the website.

Entities
- Any physical or abstract object that contains attributes and components
- i.e. receptionist, customer, user etc.
- In our website, the owner (customer), the owner's respective pet(s), as well as the Public relations communicator will be defined as entities.

Attributes
- Attributes are what defines a respective entity/entities.
- i.e. a user can have an id, name, date of birth, phone number, and address.
- Our attributes for our website will be:
  -user information, defined as user (Username, first name, middle name [optional], last name, date of birth, phone number, email, pronouns, location, biography, type[premium or normal], achievements)
  -pet information, defined as pet (Name, birthday, orientation, species, breed, biography) and public relation contact information (name, phone number, email).

Items
- Items are physical or abstract objects that users/customers can have ownership to.
- i.e. a physical/virtual product that can be purchased by the customer.
- The items that will be implemented into our website include: a virtual premium membership in which the user can purchase in the form of monthly/annual subscription

and a virtual achievement/ reward for completing certain tasks such as participating in dating other animals or meeting up to a certain number of pet owners.

Entity Relationship Diagram (ERD)

- A diagram that shows the schematics of a database for a given database system.
- i.e. An Entity Relationship Diagram of a student course registration database system created in Draw.io.
- Our website will be including an efficiently structured Entity Relationship Diagram to provide a well documented layout for constructing our Enhanced Entity Relation Diagram.

Enhanced Entity Relation Diagram (EER)

- An advanced form of entity relationship diagram that serves as the model for all the functionalities of a database system.
- i.e. An Advanced Entity Relationship Diagram of a student course registration database system created in mySQL.
- Our website will include an Enhanced Entity Relationship Diagram to provide users a high performance experience in relation to information and data storage.

Hash Tables

- A form of implementation of an associative array abstract data type, a structure that can map keys to values.
- Hash tables will be implemented into our website to store user information and record.

Linked List

- A Linked List is an ordered set of data elements that each contains a link to its respective succeeding element.
- Linked lists will be established into our website to sort the priority list of all users in order for them to be paired with the right partner on their respective date.

Stacks

- A stack is an abstract data type that holds an ordered, linear sequence of items. A stack is a last in, first out (LIFO) structure.
- Stacks can be implemented to efficiently provide users the last person they have contacted or swiped right (open to pet-dating) to.

Queues

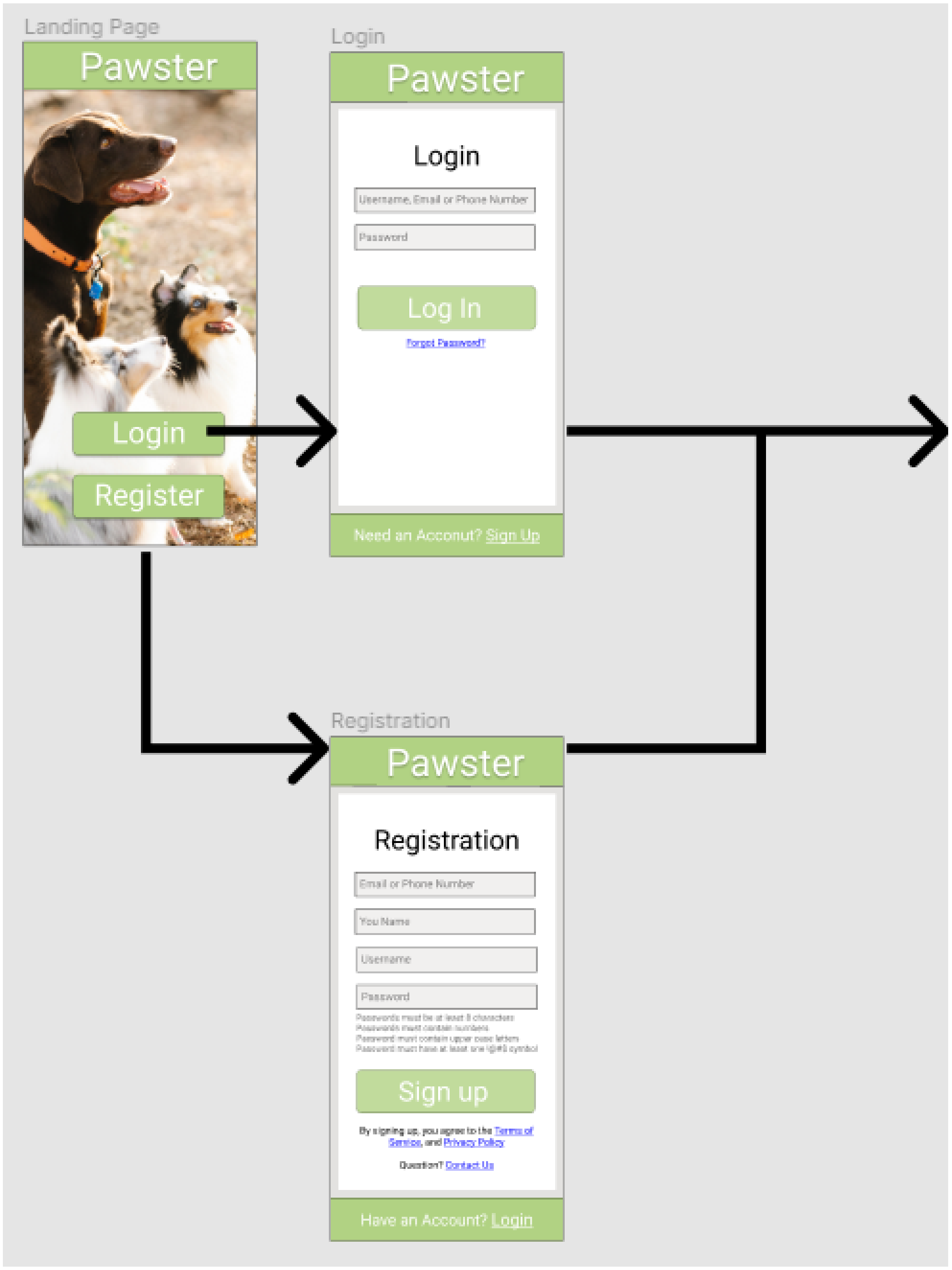
- A Queue is an abstract data type that holds an ordered, linear sequence of items. In contrast to a stack, a Queue is a first in, last out (FIFO) structure.
- Queues can be used in our website to actively show us consistent updates of user activity in an orderly fashion.

2. Functional Requirements V2

- Landing page - 3
  - Login - 1
  - Register - 1
- Login Page - 5
  - Login (username and password) option 1
  - Forget password option 1
  - Sign up option 1
- Contact page 1
  - Users should be able to contact us and include contact information. 1
- Like page - 7
  - Like button 1
  - Dislike button 1
- Profile page - 6
  - Profile image 1
  - Profile description (name, personality) textbox 1
  - Edit Profile option 1
- Sign up page - 4
  - Email textbox 1
  - Name textbox 1
  - Username textbox 1
  - Password textbox 1
  - Sign up option 1

        - Terms of policy option 1
        - Contact us option 2
    - Messaging page - 8
        - Messages 1
        - New message button 1
    - Block option - 9
        - Block user 1
        - Block confirmation message + button 1
    - Edit profile page - 7
        - Dogs Name textbox 1
        - Zip code textbox 1
        - Dogs size textbox 1
        - Interest textbox 1
        - Upload image textbox 1
        - Update profile button 1
    - FAQ page - 2
        - Website can display a list of FAQ(Frequently Asked Questions) 1

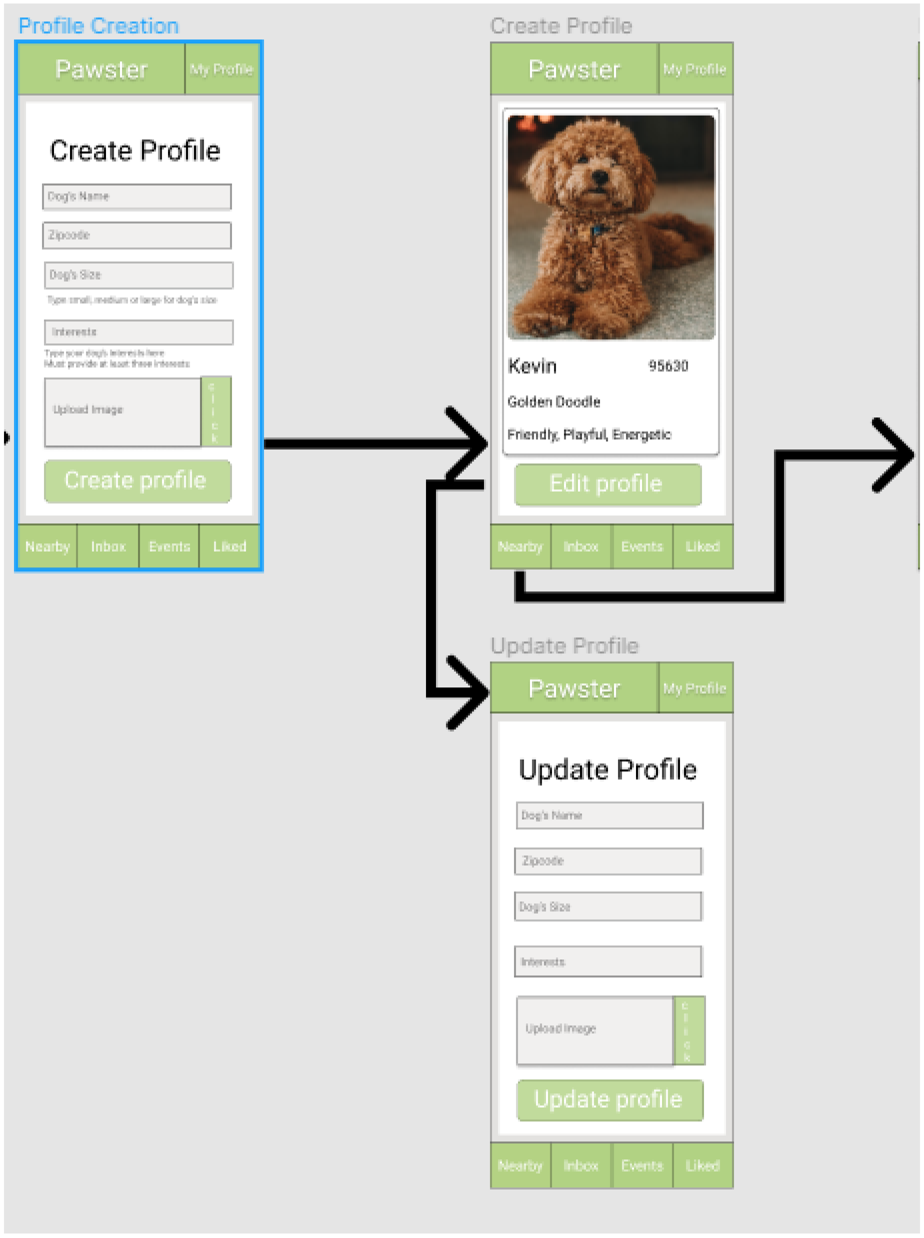
3. UI Mockups and Storyboards (high level only)
Henry Lam

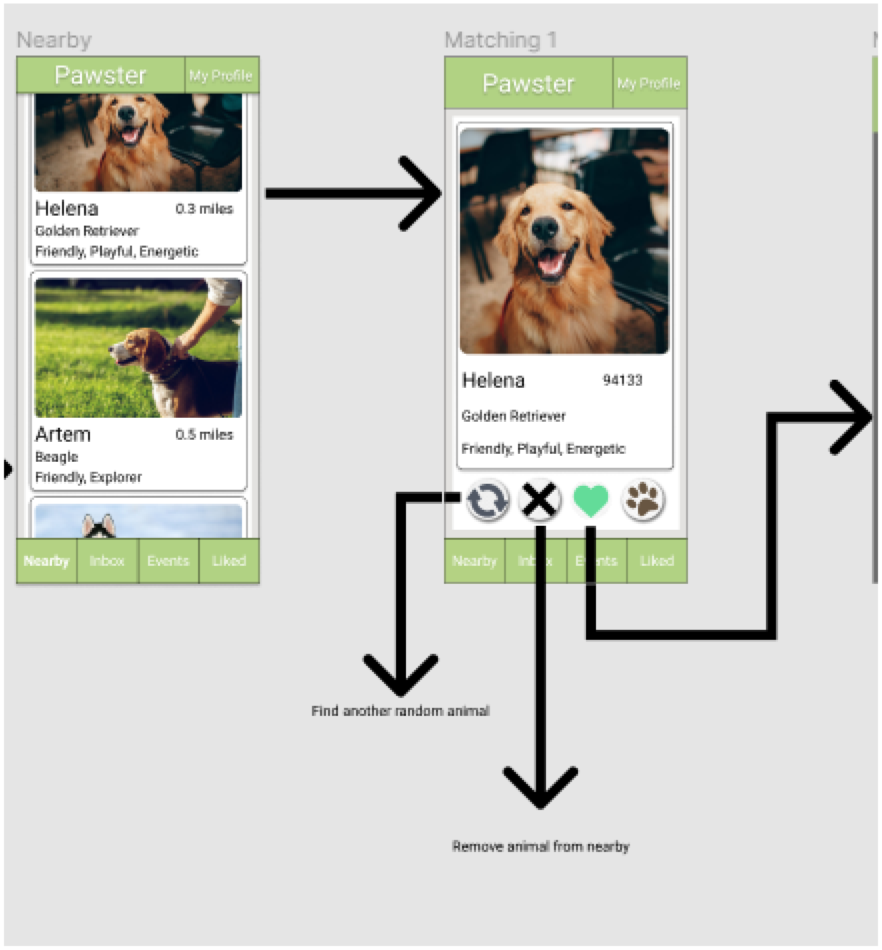


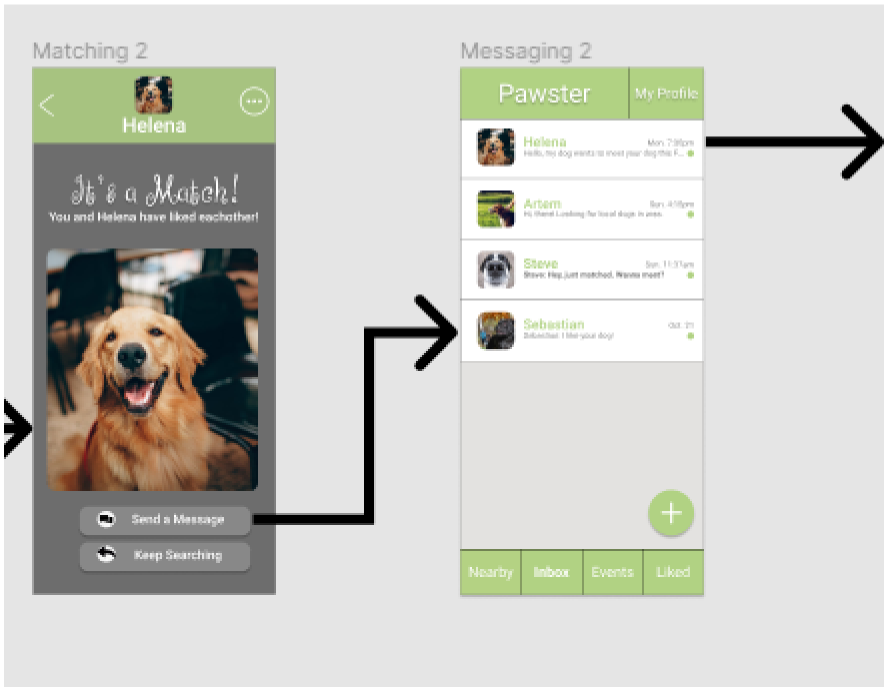
Passwords must be at least 8 characters
Passwords must contain numbers
Password must contain upper case letters
Password must have at least one !@#$ symbol

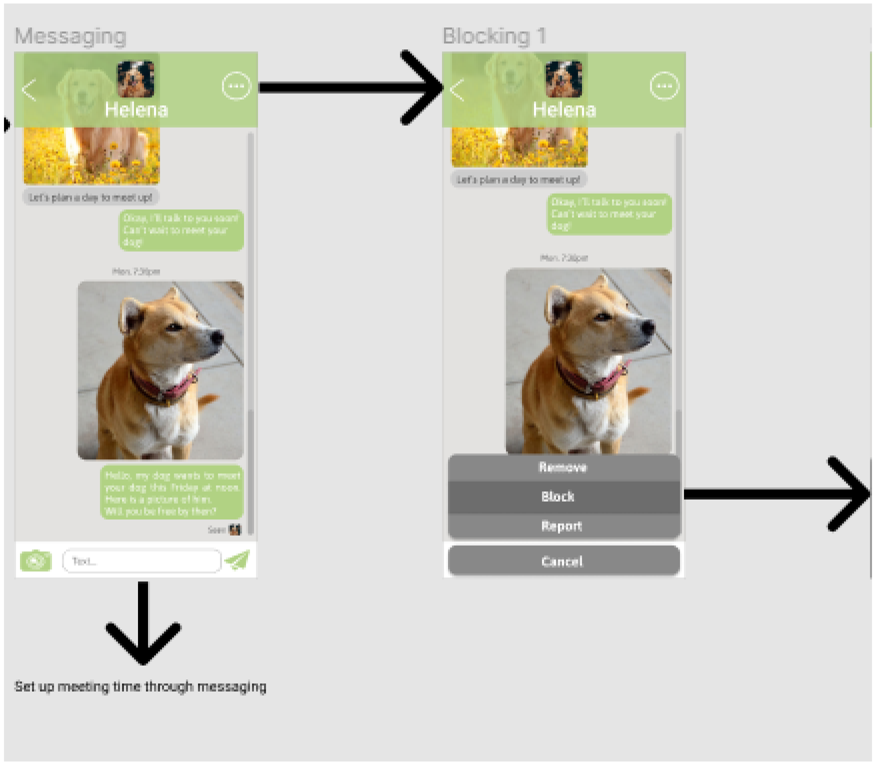
By signing up, you agree to the Terms of Service, and Privacy Policy

Question? Contact Us


Profile Creation
Pawster
My Profile
Create Profile
Dog's Name
Zipcode
Dog's Size
Type small, medium or large for dog's size
Interests
Type your dog's interests here
Must provide at least three interests
Upload Image
click
Create profile
Nearby
Inbox
Events
Liked
Create Profile
Pawster
My Profile
Kevin
95630
Golden Doodle
Friendly, Playful, Energetic
Edit profile
Nearby
Inbox
Events
Liked
Update Profile
Pawster
My Profile
Update Profile
Dog's Name
Zipcode
Dog's Size
Interests
Upload Image
click
Update profile
Nearby
Inbox
Events
Liked


Nearby
Pawster
My Profile
Helena
0.3 miles
Golden Retriever
Friendly, Playful, Energetic
Artem
0.5 miles
Beagle
Friendly, Explorer
Nearby
Inbox
Events
Liked
Matching 1
Pawster
My Profile
Helena
94133
Golden Retriever
Friendly, Playful, Energetic
Nearby
Inbox
Events
Liked
Find another random animal
Remove animal from nearby


Matching 2
Helena
It's a Match!
You and Helena have liked eachother!
Send a Message
Keep Searching
Messaging 2
Pawster
My Profile
Helena
Mon. 7:30pm
Hello, my dog wants to meet your dog this F...
Artem
Sun. 4:15pm
Hi there! Looking for local dogs in area.
Steve
Sun. 11:37am
Steve: Hey, just matched. Wanna meet?
Sebastian
Oct. 21
Sebastian: I like your dog!
Nearby
Inbox
Events
Liked



Let's plan a day to meet up!

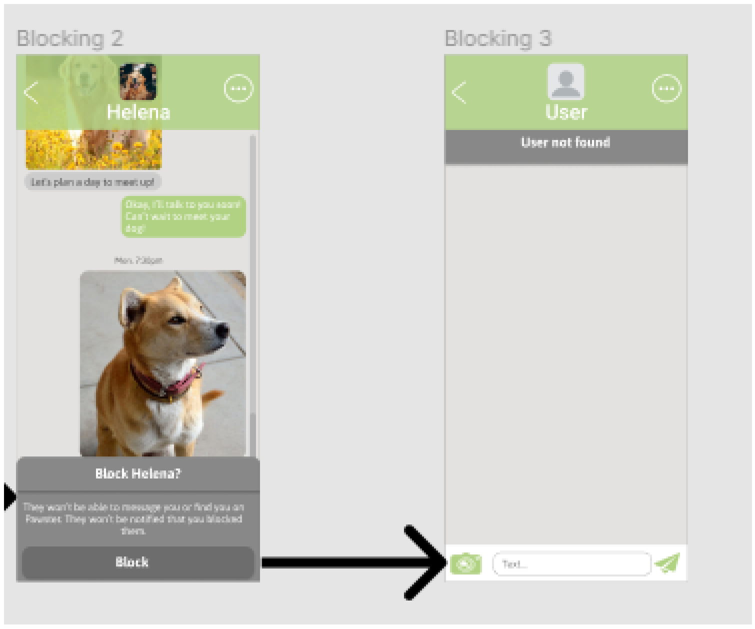
Okay, I'll talk to you soon! Can't wait to meet your dog!

Mon. 7:30pm

Hello, my dog wants to meet your dog this Friday at noon. Here is a picture of him. Will you be free by then?

Seen

Text...



Block Helena?

They won't be able to message you or find you on Pawmee. They won't be notified that you blocked them.

Block

4. High level Architecture, Database Organization
Umar Rama

- Pawster app has User, SignUp, SignIn, Pet Profile, and Inbox as entities.
- **User** has user_id as the primary key.
- **SignUp** has signUp_id as primary key, name, username, password, and email.
- **SignIn** has signIn_id as primary key, username, password, and email.
- **Pet Profile** has profile_id as primary key, name, interest, size, and image.
- **Inbox** has inbox_is as primary key, and message.

- A user can chat with multiple users.
- A user can sign up multiple times using different emails.
- A user can login/logout multiple times.
- A user can create multiple pet profiles.
- A user can update/delete pet profiles multiple times.

Conceptual Schema:
Entities: User, SignUp, SignIn, Pet Profile, Inbox.
Relationships: Register, Log In, Creates, Update/Delete, Chats.
Attributes:
- **User -** user_id.
- **SignUp -** signUp_id, name, username, password, email.
- **SignIn -** signIn_id, username, password, email.
- **Pet Profile -** profile_id, name, interest, size, image.
- **Inbox -** inbox_id, message.

Relational Schema:
- **User** (user_id: CHAR(10))
- **SignUp** (signUp_id: CHAR(10), name: VARCHAR, username: VARCHAR, password: VARCHAR, email: VARCHAR(255))
- **SignIn** (signIn_id: CHAR(10), username: VARCHAR, password: VARCHAR, email: VARCHAR(255))
- **Pet Profile** (profile_id: CHAR(10), name: VARCHAR, interest: VARCHAR, size: INT, image: BLOB)
- **Inbox** (inbox_id: CHAR(10), message: VARCHAR)
- **Register** (user_id: CHAR(10), signUp_id: CHAR(10))
- **Log In** (user_id: CHAR(10), signIn_id: CHAR(10))
- **Creates** (user_id: CHAR(10), profile_id: CHAR(10))
- **Update/Delete** (user_id: CHAR(10), profile_id: CHAR(10))
- **Chats** (user_id: CHAR(10), inbox_id: CHAR(10))

APIs

Sign Up:

- This API is used to create a new User Registration.
- If the User has already created their global account but is now creating an account for an Application, this is the API they will use to create the new account. The user id being registered on the URI must be specified. The Id of the Application the User is registering for is sent in the request body.

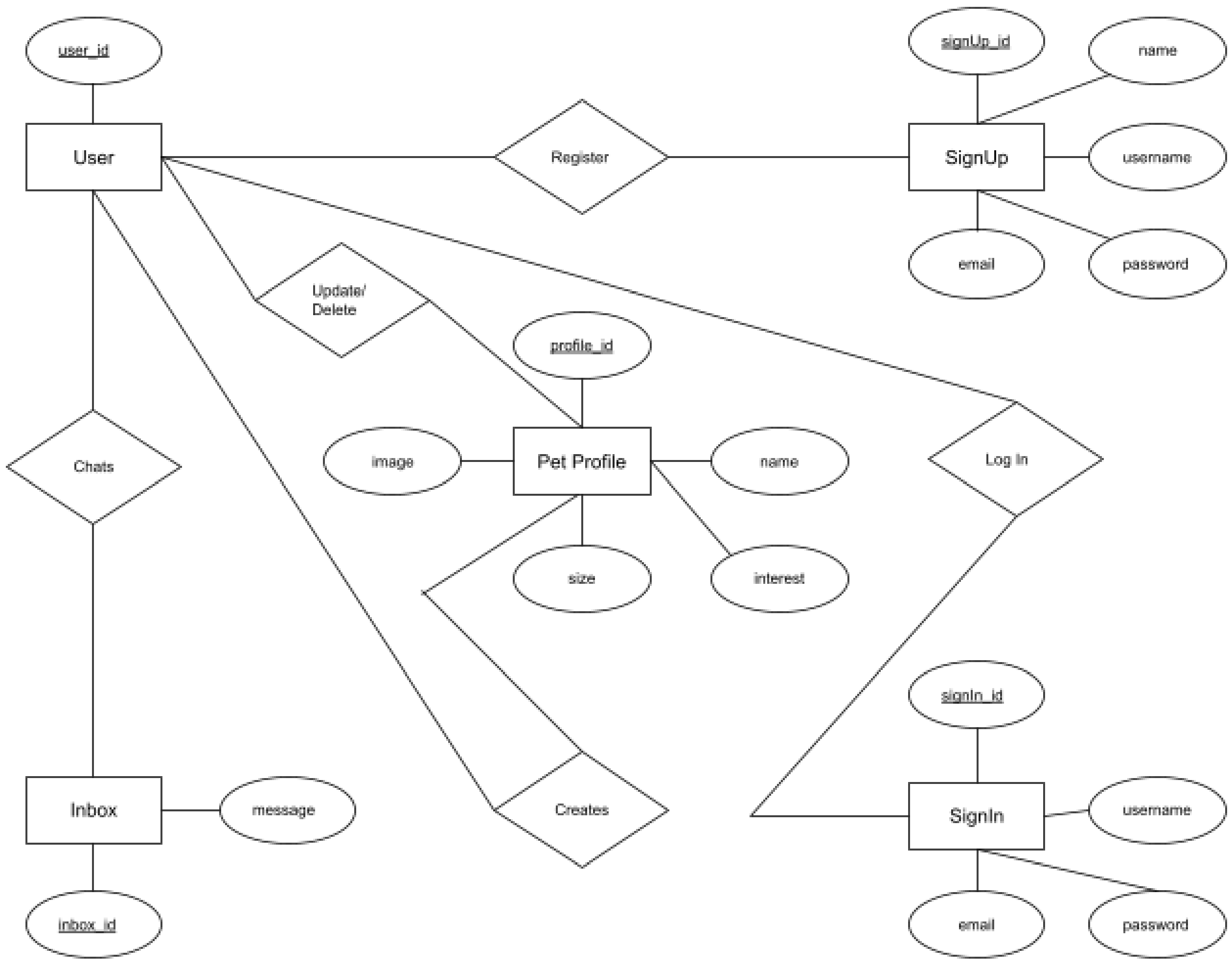
Sign In:

- This API is used to login a User.
- It is a form of API authentication that gives applications with the ability to communicate with API server to provide access. When a user logs into the system, it requests authentication in the form of a token.

Pet Profile:

- The Pet Profile API serves Amplitude pet profiles, which includes pet properties, computed pet properties, a list of cohort IDs of cohorts that the pet is in, and recommendations.
- The Profile API can be used to query identities, pet attributes and other data available on Pet Profile, anywhere it can make an HTTP request. The Profile API supports API Key authentication by setting an Authorization header.

5. High Level UML Diagrams
Umar Rama



6 .Identify actual key risks for your project at this time

At this time, there are various key points which pose large risks for the success of our application. Much of these risks have to do with the amount of functional requirements, as well as another important point that must be addressed- namely, the user base. In order for the true success of this application, we need to have a generous amount of users in order for this application to gain any success, which means that our application will not work if there aren't at least two users that are active. Since our application is meant to be used to locate other people who register for our website, most likely the smartest thing to do would be to have a small period where users can register early for our application while it’s still in development or in a beta stage, and then opening up the application when it’s ready (in a reasonable amount of time) to ensure that users can find each other when using our application.

To touch on the first point again, our functional requirements require a number of key features to implement, all of which will take their own time and will require a significant amount of programming hours. Firstly, we need to develop, or find an open source matching algorithm of some kind that can put two users together, based on information such as locational data, in order to make sure that the users themselves can meet within a reasonable margin of space. Allowing users to choose exactly how much distance they are comfortable with, as well as any other potential categories such as type of pet, general preferred meeting time, or other properties are endless, which means that constant improvements to our application will be tantamount to its success. Another issue comes from not only having to implement this algorithm, but to also build an instant messaging feature as well, which on top of implementing the user matching and registration system, will take a significant amount of work to pull off all in conjunction. Creating things such as user profiles and possibly making matches based on preferred “tags” and blocking other users is a feature that must be built, which is a very real pain point we must address.

Lastly, as our application is based off of an already popular mobile application in which users interact with the application via a smart device, designing our web application in a way that fits and is comfortable for mouse and keyboard users is a very big part of our process. Thankfully, as many applications that feature a “swipe left and right” to match or not match with other users also have web applications of their own, we have a lot of reference material to work with that tells us what is and is not feasible.

7. Project management

Haozhan Li

My team is using discord for task management. The tasks are broken up between Front end, back end. Each one can get their own task, we used google docs to track what each one is doing. Everyone can know and put down their work. Usually, we have at least one meeting each week to discuss the tasks in discord. We report each own situation to all, and will try to help the other one if someone finishes their work. In addition, we also communicate in discord chat. Through the use of discord, any team member can quickly communicate with all other members, allowing share screen transfer issues and information between team members. As of now, these project management help our team to steadily build the project.